

# Irregular turbo code design for the binary erasure channel

Ghassan M. Kraidy, Valentin Savin  
CEA-LETI, 17 rue des Martyrs, 38054 Grenoble, France  
{ghassan.kraidy,valentin.savin}@cea.fr

**Abstract**—In this paper, the design of irregular turbo codes for the binary erasure channel is investigated. An analytic expression of the erasure probability of punctured recursive systematic convolutional codes is derived. This exact expression will be used to track the density evolution of turbo codes over the erasure channel, that will allow for the design of capacity-approaching irregular turbo codes. Next, we propose a graph-optimal interleaver for irregular turbo codes. Simulation results for different coding rates is shown at the end.

## I. INTRODUCTION

The performance of error correcting codes over the binary erasure channel (BEC) can be analyzed precisely, and a flurry of research papers have already addressed this issue. For small to medium codeword length, *Maximum-Distance Separable* (MDS) codes achieve the capacity of the BEC. However, for large block lengths, their decoding becomes untractable, and thus iteratively decoded graph-based codes present the main alternative. Low-density parity-check (LDPC) codes [1] [2] [3] and repeat-accumulate (RA) codes [4] with message-passing decoding proved to perform very close to the channel capacity with reasonable complexity. Moreover, “rateless” codes [5] [6] that are capable of generating an infinite sequence of parity symbols were proposed for the BEC. However, convolutional-based codes, that are widely used for Gaussian channels, are less investigated on the BEC. Among the few papers that deal with convolutional and turbo codes over the BEC are [7] [8] [9] [10]. In this paper, we propose irregular turbo codes that approach the capacity of the BEC for medium to large block length. This is accomplished through precise asymptotic analysis of the codes together with a graph-optimal interleaver. The paper is organized as follows: in Section II we describe the model of the irregular turbo code. Section III gives the exact erasure probability at the output of a punctured RSC code. The asymptotic design of irregular turbo codes is then discussed in Section IV, while Section V presents an optimal graph-based interleaver for such codes. Section VI shows the performance of these codes and Section VII gives the concluding remarks.

## II. IRREGULAR TURBO CODES

A parallel turbo code [11] generally consists of a concatenation of two recursive systematic convolutional (RSC) codes. An information sequence  $\mathbf{b}$  is encoded by the first RSC code to generate a first parity bit sequence; the same sequence is then scrambled by an interleaver  $\Pi$  and encoded by a second RSC code to generate a second parity bit sequence. In most cases, the two constituent RSC encoders of a parallel turbo code are identical. For this reason, the authors in [12] [13]

proposed a “self-concatenated” turbo encoder in which every information bit is repeated twice, interleaved, and fed to an RSC code of double the size, as shown in Fig. 1. In this

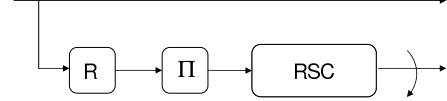


Fig. 1. Self-concatenated turbo encoder.

new representation, each information bit is connected to the code trellis via two edges in the propagation tree of Fig. 2. Therefore, we say that the *degree* of the information bits is  $d = 2$ , and that the turbo code is regular. Using this structure, one can create irregularity by repeating a certain fraction  $f_d$  of the bits  $d$  times, providing bits that are more protected than in the regular case. Like for LDPC and RA codes [2] [14], irregularity can boost the performance of turbo codes for large block lengths. Irregular turbo codes were first introduced in [15]. In [12] [13], in a slightly different design, a fraction of the information bits is repeated  $d$  times with  $d > 2$ , while the parity bits remained of degree 1. In order to maintain the same coding rate, a fraction  $\phi_p$  of the parity bits is punctured. We will use this representation to design irregular turbo codes for the binary erasure channel. The encoder of an irregular turbo code is similar to that of Fig. 1, with the difference that the repetition is non-uniform. The information bits are thus divided into  $d$  classes with  $d = 2, \dots, d_{\max}$ , where  $d_{\max}$  is the maximum bit-node degree. The number of bits in a class  $d$  is a fraction  $f_d$  of the total number of information bits at the turbo encoder input, where bits in class  $d$  are repeated  $d$  times. Finally, the output of the non-uniform repeater is interleaved and fed to the RSC constituent code, of which  $(1 - \phi_p)$  of the parity bits are transmitted. Now let  $K$  denote the length of the information sequence,  $N$  the interleaver size,  $\rho_0$  and  $\rho$  the initial and the final (punctured) rate of the RSC constituent code respectively, and  $R_c$  the rate of the turbo code. We can write the following:

$$\sum_{d=2}^{d_{\max}} f_d = 1, \quad \sum_{d=2}^{d_{\max}} d \cdot f_d = \bar{d}, \quad N = K \sum_{d=2}^{d_{\max}} d \cdot f_d = K \cdot \bar{d} \quad (1)$$

$$R_c = \frac{K}{K + \frac{N}{\rho} - N} = \frac{1}{1 + \left(\frac{1}{\rho} - 1\right) \bar{d}} \quad (2)$$

$$\rho = \frac{1}{1 + (1 - \phi_p) \left(\frac{1}{\rho_0} - 1\right)} \quad (3)$$

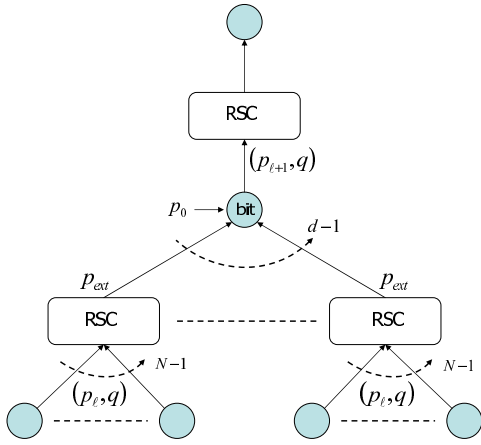


Fig. 2. Propagation tree of an irregular turbo code

For a degree profile  $\{f_2, f_3, \dots, f_{d_{\max}}\}$  and using the above equations, one can compute the puncturing fraction  $\phi_p$  corresponding to a target rate  $R_c$ . The performance of an irregular turbo code will strongly depend on the degree profile and the puncturing fraction or, more specifically, on the corresponding puncturing pattern. In the following sections we will consider the design of capacity-approaching irregular turbo codes over the BEC. To do so, we will first compute the analytic expression of the extrinsic erasure probability at the output of the punctured RSC decoder that represents the key tool for the density evolution of irregular turbo codes.

### III. ERASURE PROBABILITY OF PUNCTURED RSC CODES

In this section, we will derive the exact erasure probability of binary RSC codes, taking into account the puncturing of parity bits. To do so, we will follow the steps of the method proposed in [7] used to compute the erasure probability at the output unpunctured RSC codes. For the sake of simplicity, we only consider half-rate codes with constraint length  $L = \nu + 1$ , where  $\nu$  is the memory of the code. The same method applies to RSC codes with different rates.

We consider the following communication scheme: a uniformly distributed sequence of bits  $\mathbf{b}$  of length  $K$  is fed to a binary RSC encoder that generates a sequence  $\mathbf{c}$  of parity bits of length  $N(1 - \phi_p)$ . During transmission<sup>1</sup>, a bit  $b_i$  (respectively  $c_j$ ) is either erased with probability  $p$  (respectively  $q$ ), or perfectly received with probability  $1 - p$  (respectively  $1 - q$ ). Let  $\mathbf{b}'$  and  $\mathbf{c}'$  be the received sequences at the decoder. An RSC code has  $S = 2^\nu$  states. Considering the “Forward-Backward” [16] decoding algorithm, let  $F_n(s)$  and  $B_n(s)$  be the probabilities of being in state  $s = 1, \dots, S$  computed in the forward and in the backward directions, at the left and at the right side of the  $n^{\text{th}}$  trellis step respectively. Let  $l(e)$  and  $r(e)$  be the states to which an edge  $e$  is connected

<sup>1</sup>We consider different erasure probabilities on information and parity bits, in order to be able to distinguish between the extrinsic (corresponding to information bits) and the communication (corresponding to parity bits) channels for the density evolution computation

on the left and on the right respectively. The information bit  $b(e)$  and parity bit  $c(e)$  are associated to edge  $e$ . As shown in [7], the extrinsic probability of an information bit  $b_n$  at the output of the decoder is written as:

$$P_{\text{ext}}(b_n) = P(b_n | \mathbf{b}'_{-\infty}^{n-1}, \mathbf{b}'_{n+1}^{\infty}, \mathbf{c}') \propto \sum_{e: b(e)=b_n} F_n(l(e)) \cdot P(c(e)) \cdot B_n(r(e)) \quad (4)$$

Now let  $\Sigma_F = \{\sigma_f^1, \dots, \sigma_f^{|\Sigma_F|}\}$  and  $\Sigma_B = \{\sigma_b^1, \dots, \sigma_b^{|\Sigma_B|}\}$  be the sets from which  $F_n$  and  $B_n$  take values. The cardinality of the sets  $\Sigma_F$  and  $\Sigma_B$  is computed as:

$$|\Sigma_F| = |\Sigma_B| = \sum_{\alpha=0}^{\nu} \binom{2^\alpha}{2^\nu} \quad (5)$$

However, as an RSC code is linear, we assume the all-zeros codeword is transmitted without losing generality. This gives smaller state distribution sets  $\Sigma_F^*$  and  $\Sigma_B^*$  with cardinality:

$$|\Sigma_F^*| = |\Sigma_B^*| = \sum_{\alpha=0}^{\nu} \binom{2^\alpha - 1}{2^\nu - 1} \quad (6)$$

A four-state RSC code ( $L = 3$ ) has for instance:

$$\Sigma_F^* = \Sigma_B^* = \{(1, 0, 0, 0), (1/2, 1/2, 0, 0), (1/2, 0, 1/2, 0), (1/2, 0, 0, 1/2), (1/4, 1/4, 1/4, 1/4)\} \quad (7)$$

#### A. Computation of the Erasure Probability

The trellis of a convolutional code forms two first-order  $S$ -state Markov chains corresponding to the forward and backward recursions. This allows to compute the steady-state distributions of the Markov processes that will be used to compute the bit erasure probability at the output of the decoder. The distributions  $\pi_F(p, q)$  and  $\pi_B(p, q)$  are the normalized solutions of the following equations:

$$\pi_F(p, q) = \pi_F(p, q) \cdot M_F(p, q); \quad \pi_B(p, q) = \pi_B(p, q) \cdot M_B(p, q) \quad (8)$$

where the  $(i, j)^{\text{th}}$  entry of matrix  $M_F$  is the probability of the transition from state distribution  $F_n = \sigma_f^i$  to state distribution  $F_{n+1} = \sigma_f^j$ . Similarly, the matrix  $M_B$  represents the transition probabilities in the backward direction. In other words, the distributions  $\pi_F$  and  $\pi_B$  are the stationary distributions to which the Markovian process converges, as:

$$\lim_{\delta \rightarrow \infty} M_F^\delta = \mathbf{1} \otimes \pi_F; \quad \lim_{\delta \rightarrow \infty} M_B^\delta = \mathbf{1} \otimes \pi_B \quad (9)$$

where  $\mathbf{1}$  is a column vector of ones. As an example, we will consider the four-state RSC  $(1, 5/7)_8$  code with  $L = 3$ . Let  $p$  be the erasure probability on the information bits, and  $q$  be the erasure probability on parity bits. Assuming the all-zeros codeword has been transmitted, we have that  $|\Sigma_F^*| = |\Sigma_B^*| = 5$ . The  $5 \times 5$  Markov state transition matrix  $M_F$  for the forward recursion of this code is given by:

$$M_F(p, q) = \begin{bmatrix} 1-pq & 0 & pq & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1+pq-p-q & p-pq & 0 & q-pq & pq \\ 1+pq-p-q & q-pq & 0 & p-pq & pq \\ 0 & 0 & 1+pq-p-q & 0 & p+q-pq \end{bmatrix}$$

and the matrix  $M_B$  for the backward recursion is given by:

$$M_B(p, q) = \begin{bmatrix} 1-pq & pq & 0 & 0 & 0 \\ 1+pq-p-q & 0 & p-pq & q-pq & pq \\ 0 & 1 & 0 & 0 & 0 \\ 1+pq-p-q & 0 & q-pq & p-pq & pq \\ 0 & 1+pq-p-q & 0 & 0 & p+q-pq \end{bmatrix}$$

Once  $M_F(p, q)$  and  $M_B(p, q)$  are computed, we can solve for  $\pi_F(p, q)$  and  $\pi_B(p, q)$ . We next consider the matrix  $T(q)$  whose  $(i, j)^{\text{th}}$  entry represents the probability of an output erasure conditioned on the left and right state distributions  $\sigma_f^i$  and  $\sigma_b^j$ , knowing that parity bits are erased with probability  $q$ :

$$T_{i,j}(q) = P(P_{\text{ext}}(b_n) = 1/2 \mid F_n = \sigma_f^i, B_n = \sigma_b^j) \quad (10)$$

The matrix  $T(q)$  for the RSC  $(1, 5/7)_8$  code is given by:

$$T(q) = \begin{bmatrix} 0 & 0 & q & 0 & 0 \\ q & q & q & q & q \\ 0 & 1 & q & 0 & 1 \\ 0 & 0 & q & 1 & 1 \\ q & 1 & q & 1 & 1 \end{bmatrix}$$

Finally the extrinsic erasure probability is computed as:

$$P_{\text{ext}}(p, q) = \pi_F(p, q) \cdot T(q) \cdot \pi_B(p, q)^t \quad (11)$$

where the operator  $(\cdot)^t$  denotes the transpose operator.

### B. Computation of the Erasure Probability with puncturing

Now suppose a fraction  $\phi_p$  of the parity bits of the code are punctured. If the punctured parity bits were randomly chosen at each transmission, we could consider that the fraction  $\phi_p$  of punctured bits is a part of the channel, as if the decoder receives bits with probability of erasure on parity bits given by:

$$q' = 1 - (1 - q)(1 - \phi_p) = \phi_p + q - q \cdot \phi_p \quad (12)$$

However, if the puncturing pattern is fixed, the extrinsic erasure probability computed using (11) by replacing  $q$  with  $q'$  from (12) is inaccurate. The goal is then to analytically compute the extrinsic erasure probability at the output of the decoder knowing that parity bits are punctured using a predefined pattern. For this purpose, we define a puncturing pattern  $\mathbb{X} = [x_1, x_2, \dots, x_\Gamma]$ ,  $x_\gamma \in \{0, 1\}$ , where a 0 in position  $\gamma$  means that the parity bit in the corresponding trellis step is punctured. The parity bits of the constituent RSC code are then punctured using a periodic puncturing pattern with period  $\mathbb{X}$ . We consider a window of size  $\Gamma$  in the trellis of the code, and let  $M_{F,\mathbb{X}}(p, q)$  the matrix whose  $(i, j)^{\text{th}}$  entry is the probability of the transition from the state distribution  $F_n = \sigma_f^i$  at the left side of the window to the state distribution  $F_{n+\Gamma} = \sigma_f^j$  at the right side of the window. Similarly, the matrix  $M_{B,\mathbb{X}}$  represents “throughout-the-window” transition probabilities in the backward direction. We have the following:

$$M_{F,\mathbb{X}}(p, q) = \prod_{\gamma=1}^{\Gamma} M_F(p, q^{x_\gamma}); \quad M_{B,\mathbb{X}}(p, q) = \prod_{\gamma=1}^{\Gamma} M_B(p, q^{x_{\Gamma+1-\gamma}}) \quad (13)$$

This means that  $M_{F,\mathbb{X}}(p, q)$  is obtained by multiplying matrices  $M_F(p, 1)$  and  $M_F(p, q)$  according to whether the

corresponding parity bit is punctured ( $x_\gamma = 0$ ) or not ( $x_\gamma = 1$ ). A similar assertion holds for the backward matrix  $M_{B,\mathbb{X}}(p, q)$ .

Let  $\pi_{F,\mathbb{X}}(p, q)$  and  $\pi_{B,\mathbb{X}}(p, q)$  be the corresponding steady-state distributions, meaning that:

$$\lim_{\delta \rightarrow \infty} M_{F,\mathbb{X}}^\delta = \mathbf{1} \otimes \pi_{F,\mathbb{X}}; \quad \lim_{\delta \rightarrow \infty} M_{B,\mathbb{X}}^\delta = \mathbf{1} \otimes \pi_{B,\mathbb{X}} \quad (14)$$

where  $\mathbf{1}$  is a column vector of ones. These expressions represent the state probability distributions in the forward and backward directions, at the left and at the right side of the window respectively. The distributions  $\pi_{F,\gamma}(p, q)$  on the left side and  $\pi_{B,\gamma}(p, q)$  on the right side of a window step  $\gamma$  can be recursively computed as:

$$\begin{aligned} \pi_{F,1}(p, q) &= \pi_{F,\mathbb{X}}(p, q), \\ \pi_{F,\gamma}(p, q) &= \pi_{F,\gamma-1}(p, q) \cdot M_F(p, q^{x_\gamma}), \quad \gamma = 2, \dots, \Gamma \end{aligned} \quad (15)$$

$$\begin{aligned} \pi_{B,\Gamma}(p, q) &= \pi_{B,\mathbb{X}}(p, q), \\ \pi_{B,\gamma}(p, q) &= \pi_{B,\gamma+1}(p, q) \cdot M_B(p, q^{x_\gamma}), \quad \gamma = \Gamma - 1, \dots, 1 \end{aligned} \quad (16)$$

Next, the extrinsic erasure probability of the information bit in position  $\gamma$  can be computed as:

$$P_{\text{ext},\gamma}(p, q) = \pi_{F,\gamma}(p, q) \cdot T(q^{x_\gamma}) \cdot \pi_{B,\gamma}(p, q)^t \quad (17)$$

Finally, the extrinsic erasure probability at the output of the decoder corresponding to the puncturing pattern  $\mathbb{X}$  is given by:

$$P_{\text{ext},\mathbb{X}}(p, q) = \frac{1}{\Gamma} \sum_{\gamma=0}^{\Gamma} P_{\text{ext},\gamma}(p, q) \quad (18)$$

As an example, suppose we want to construct a half-rate parallel turbo code using half-rate  $(1, 5/7)_8$  RSC codes. In order to raise the rate of the constituent codes from  $1/2$  to  $2/3$ , we puncture half of their parity bits using the pattern  $\mathbb{X} = [1, 0]$ . The expression of the exact probability of this code can then be written as:

$$P_{\text{ext},\mathbb{X}}(p, q) = \frac{1}{2} [P_{\text{ext},1}(p, q) + P_{\text{ext},2}(p, q)] \quad (19)$$

where

$$P_{\text{ext},1} = \pi_{F,\mathbb{X}}(p, q) \cdot T(q) \cdot [\pi_{B,\mathbb{X}}(p, q) \cdot M_B(p, 1)]^t \quad (20)$$

$$P_{\text{ext},2} = [\pi_{F,\mathbb{X}}(p, q) \cdot M_F(p, q)] \cdot T(1) \cdot \pi_{B,\mathbb{X}}(p, q)^t \quad (21)$$

$$\begin{aligned} \pi_{F,\mathbb{X}}(p, q) &= \frac{1}{5} \cdot \mathbf{1}^t \cdot \lim_{\delta \rightarrow \infty} M_{F,\mathbb{X}}^\delta(p, q), \text{ and} \\ M_{F,\mathbb{X}}(p, q) &= M_F(p, q) \cdot M_F(p, 1) \end{aligned} \quad (22)$$

$$\begin{aligned} \pi_{B,\mathbb{X}}(p, q) &= \frac{1}{5} \cdot \mathbf{1}^t \cdot \lim_{\delta \rightarrow \infty} M_{B,\mathbb{X}}^\delta(p, q), \text{ and} \\ M_{B,\mathbb{X}}(p, q) &= M_B(p, 1) \cdot M_B(p, q) \end{aligned} \quad (23)$$

The exact expression of the erasure probability in (18) is the key tool for designing irregular turbo codes for the BEC, as will be discussed in the following section. In fact, for the same puncturing fraction  $\phi_p$ , it is capable of determining which pattern  $\mathbb{X}$  gives the lowest  $P_{\text{ext},\mathbb{X}}$ . Moreover, it allows to detect a catastrophic puncturing scenario that leads to infinite error events and thus harms the correction capacity of the code. As an example, puncturing the RSC  $(1, 5/7)_8$  code using  $\mathbb{X} = [1, 0, 0]$  gives:

$$P_{\text{ext},\mathbb{X}}(0, q) > 0 \quad (24)$$

This means that this puncturing pattern is catastrophic, as a single bit error at the input of the decoder generates an infinite error event. If we have  $\phi_p = 2/3$ , we would rather use  $\mathbb{X} = [1, 0, 0, 0, 1, 0]$  for instance. Although this example can be directly observed on the trellis of the  $(1, 5/7)_8$  code, (18) points out the phenomenon for any  $S$ -state code (on any channel!), where trellis analysis becomes more tedious as  $S$  increases.

#### IV. IRREGULAR TURBO CODE DESIGN

The analytic expression of the erasure probability of punctured RSC codes in the previous section allows us to analyze the iterative decoding of turbo codes over the BEC. As discussed in Section II, a parallel turbo code consists of a parallel concatenation of two RSC codes. The iterative decoding of such codes can be analyzed through EXIT charts [17], that are non-linear functions relating the output to the input of the RSC decoders of the infinite-length turbo code. This technique gives insight on the iterative process in the sense that the decoding is successful for a certain channel quality if the two curves corresponding to the two decoders do not intersect. The threshold of the code is the worst value of the channel quality at which the tunnel between the two curves is open. In the case where the two constituent codes are identical, the decoding converges if the curve of the RSC decoder does not intersect with the forty-five degree line. Over the BEC, and with the difference of Gaussian channels in general, an EXIT chart describing the iterative decoding process gives the exact density evolution of erasure probabilities, as we can compute analytic expressions of the output as a function of the input of the decoder. Although widely used for LDPC codes, this property was first exploited in [18] to compute exact thresholds for regular unpunctured turbo codes. For the sake of infinite-length analysis, we represent a turbo code using the tree structure as shown in Fig. 2, in which an information bit of degree  $d$  is connected to  $d$  trellises. For the regular parallel turbo code, the erasure probability at iteration  $\ell + 1$  is given by:

$$P_{\ell+1} = p_0 \cdot P_{\text{ext},\mathbb{X}}(P_\ell, p_0) \quad (25)$$

where  $p_0$  is the channel erasure probability. This expression determines the density evolution of the iterative decoding process, as it relates the probability at an iteration to that of the previous iteration. Using (25), the threshold probability  $p_{\text{th}}$  of the  $R_c = 1/3$  parallel turbo code built from rate-half RSC  $(1, 5/7)_8$  constituent codes is computed as  $p_{\text{th}} = 0.6428$ , knowing that the capacity of the BEC is  $C = 1 - p_0$ . Again, the punctured  $R_c = 1/2$  turbo code built from the same constituent codes has  $p_{\text{th}} = 0.4729$ .

In order to tighten the gap to the capacity of the BEC at a given rate, we consider the design of irregular turbo codes. The erasure probability at iteration  $\ell + 1$  of a bit of degree  $d$  can be expressed as a function of the erasure probability at iteration  $\ell$  as:

$$P_{\ell+1}(d) = p_0 \cdot \frac{d \cdot f_d}{d} \cdot P_{\text{ext},\mathbb{X}}(P_\ell, p_0)^{d-1} \quad (26)$$

Let  $\lambda_d = \frac{d \cdot f_d}{d}$  and  $\lambda(X) = \sum_{d=1}^{d_{\max}} \lambda_d X^{d-1}$  (this definition will be made clearer in Section V where we will introduce the factor graph of the turbo code). We can then write:

$$P_{\ell+1}(d) = p_0 \cdot \lambda_d \cdot P_{\text{ext},\mathbb{X}}(P_\ell, p_0)^{d-1} \quad (27)$$

Averaging over all possible bit degrees, we get:

$$P_{\ell+1} = p_0 \cdot \lambda \circ P_{\text{ext},\mathbb{X}}(P_\ell, p_0) \quad (28)$$

Following this equation, the irregular turbo code can recover from a channel erasure probability  $p_0$  if and only if

$$p_0 \cdot \lambda \circ P_{\text{ext},\mathbb{X}}(x, p_0) \leq x, \quad \forall x \in [0, p_0] \quad (29)$$

The code threshold is defined as:

$$p_{\text{th}}(\lambda, \mathbb{X}) = \max\{p_0 \mid p_0 \cdot \lambda \circ P_{\text{ext},\mathbb{X}}(x, p_0) \leq x, \quad \forall x \in [0, p_0]\} \quad (30)$$

and it depends on both degree distribution and puncturing pattern. The design of capacity approaching irregular turbo codes reduces to the optimization of the function  $(\lambda, \mathbb{X}) \mapsto p_{\text{th}}(\lambda, \mathbb{X})$ . For instance, this can be carried out using the differential evolution algorithm [19]. In general, a uniform puncturing pattern leads to the best threshold, provided the pattern is not catastrophic (which leads to a threshold equal to zero!). Therefore, in order to reduce the space of parameters of the optimization function, for each degree distribution  $\lambda$ , we compute the puncturing fraction  $\phi_p$  according to the target rate  $R_c$ , and chose the puncturing pattern  $\mathbb{X}$  as uniform as possible according to  $\phi_p$ . As an example, using the half-rate RSC  $(1, 5/7)_8$  code and by setting  $d_{\max} = 12$ , we obtained the degree profiles in Table I.

TABLE I  
DEGREE PROFILE OF IRREGULAR TURBO CODES OVER THE BEC

$R_c$	$f_2$	$f_4$	$f_5$	$f_7$	$f_8$	$f_9$	$f_{12}$	$\bar{d}$	$\phi_p$	$p_{\text{th}}$
1/2	0.801	0.101			0.046		0.052	2.998	0.666	0.490
1/3	0.838		0.034	0.041	0.042	0.045		2.873	0.304	0.665
1/4	0.837		0.055		0.054		0.054	3.033	0.011	0.743

Note that the half-rate irregular turbo code designed through differential evolution has 2 parity bits out of 3 punctured, and the optimization algorithm avoided catastrophic puncturing while computing (18), as explained at the end of Section III.

#### V. PEG-BASED INTERLEAVER FOR TURBO CODES

We investigate now the design of graph-based interleavers for irregular turbo codes based on the progressive-edge growth (PEG) algorithm [20]. To do so, we define the factor graph of an irregular turbo code, in a manner similar to that of [21], [22]. As shown in Fig. 3 the factor graph consists of:

- bit nodes, represented by simple circles (information bits are represented on the top, while parity bits are represented on the bottom);
- state nodes, represented by double circles;

- trellis step nodes, also called transition nodes, represented by squares.

If  $\rho_0 = k/n$  is the rate of the constituent RSC codes, then each transition node is connected to  $k$  information bits and  $n - k$  parity bits. Using this representation, the previously defined  $\lambda_d = \frac{d \cdot f_d}{d}$  is equal to the fraction of edges emanating from information bit nodes of degree  $d$ , and  $\lambda = (\lambda_2, \dots, \lambda_{d_{\max}})$  is called the *edge perspective* degree distribution.

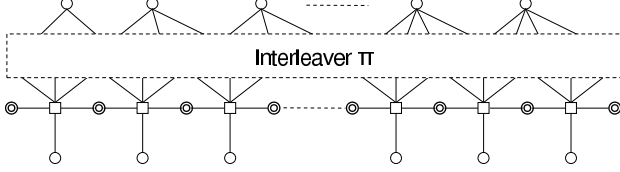


Fig. 3. Factor graph of irregular turbo codes.

The decoding of turbo codes can be performed on the factor graph, by iteratively propagating extrinsic messages from each graph node to its neighbor nodes. As discussed in [23][22][24], small cycles must be avoided in the factor graph of a turbo code so that it looks locally tree-like, and thus the messages are more independent. An upper-bound on the girth (minimal cycle length) of an irregular factor graph can be derived using a straightforward variation of the approach in [25] (see also Lemma 2 in [20]):

$$g \leq 4 \left( \left\lceil \frac{\log \left[ (N-1) \frac{k}{k+2} + 1 \right]}{\log(k+1)} \right\rceil + 1 \right) \quad (31)$$

Thus, a graph-optimal interleaving algorithm for irregular turbo codes would yield factor graphs with girths that grow as the logarithm of the interleaver size. Graphs with large girths have been already used for the construction of regular turbo codes [22] and LDPC codes [20]. The *Progressive Edge Growth* algorithm proposed in [20] is based on a simple but very efficient idea: it progressively establishes “best-effort” connections in the graph, where a best-effort connection corresponds to an edge maximizing the graph girth. In what follows, we extend this construction to the case of irregular turbo codes. The corresponding interleaver will be called PEG interleaver.

The algorithm is submitted with the set  $\mathcal{B} = \{b_1, \dots, b_K\}$  of information bit nodes, the set  $\mathcal{T} = \{t_1, \dots, t_N\}$  of transition nodes, and a desired information bit degree distribution. According to the submitted distribution, we can write  $\mathcal{B}$  as a disjoint union  $\mathcal{B} = \cup_{d=2}^{d_{\max}} \mathcal{B}_d$ , where  $\mathcal{B}_d$  is the set of information bits with submitted degree  $d$ . The algorithm starts with a factor graph comprising the set  $\mathcal{B}$  of information bit nodes, the set  $\mathcal{T}$  of transition nodes and the corresponding set of state nodes, each transition node being connected to its left and right state nodes. At this moment there is no connection between information bit and transition nodes. We then progressively add edges emanating from bits in the set  $\mathcal{B}_2$ , until all these bits reach the submitted degree 2. We next progressively connect the bits from the sets  $\mathcal{B}_3, \dots, \mathcal{B}_{d_{\max}}$ . It

is important to notice that no bit of  $\mathcal{B}_d$  is connected, as long as there are bits in  $\mathcal{B}_{d-1}$  that do not reach the submitted degree  $(d-1)$ . This is done in order to protect the bits of small degree in the following sense: when information bits of small degree (e.g.  $d=2$ ) are connected, the graph girth is relatively large; this will help to avoid short cycles that contain only information bit nodes of small degree. Adding more edges in the graph, the girth will decrease. When information bit nodes of higher degree are connected, we get smaller cycles, but these cycles are better connected to other cycles in the graph. To connect bit nodes in  $\mathcal{B}_d$  we proceed as follows:

**Progressive Edge Growth for  $\mathcal{B}_d$**   
**for**  $i = 1, \dots, d$   
  **for each**  $b \in \mathcal{B}_d$   
    **if**  $i = 1$   
      Choose a transition node  $t$  of lowest degree in the current graph and connect  $b$  to  $t$   
    **else**  
      Expand the current graph as a tree rooted at  $b$ , until all transition nodes are in the tree. Identify transition nodes that are connected to at most  $k-1$  information bits in the current graph. Among these transition nodes, identify those of maximal depth in the tree. Among these last identified transition nodes, choose a transition node  $t$  of lowest degree in the current graph and connect  $b$  to  $t$   
    **end**  
  **end**  
**end**

Note that, we first add an edge for each bit in  $\mathcal{B}_d$ , then a second edge for each bit, and so on, until all bits reach the degree  $d$ .

In case that the puncturing pattern is known at the time of the interleaver construction, we can use a slightly modified version of the above algorithm as follows: first, each bit  $b \in \mathcal{B}$  is connected to an unpunctured transition node of lowest degree in the current graph (we say that a transition node is unpunctured if its parity bit is unpunctured). In case that the number of unpunctured transition nodes is less than the number of information bit nodes, we connect the information bits of smallest submitted degree. Finally, the interleaver is constructed by running the above PEG algorithm, but starting from this graph. In this way, we make sure that if the number of unpunctured transition nodes is greater than the number of information bit nodes, then any information bit is connected to at least an unpunctured transition node.

Now that we have constructed the PEG-based interleaver, we can derive a lower-bound of the girth of the corresponding factor graph. Similar to the proof of Theorem 1 in [20], we can easily show that:

$$g \geq 2 \left( \left\lceil \frac{\log \left[ N(k+2) \left( 1 - \frac{1}{d_{\max}} \right) - N + 1 \right]}{\log [(d_{\max} - 1)(k+1)]} \right\rceil + 2 \right) \quad (32)$$

Otherwise formulated, the girth of an irregular PEG interleaved factor graph increases with the logarithm of the interleaver size, which is optimal according to (31).

Over the BEC, the performance under iterative decoding is determined by the minimum *stopping set* [1] [9] of the factor graph of the code, the size of which is upper-bounded by its minimum distance. As this minimum stopping set is in general related to the girth of a factor graph [26], we would expect its

size increases with the girth. Moreover, the minimum distance of a turbo code grows logarithmically with the interleaver size [24]. The rate of growth of the minimum stopping set of turbo codes under PEG-based interleaving would thus be optimal.

## VI. SIMULATION RESULTS

In this section, frame error rate performance of irregular turbo codes over the BEC is shown. Although the analysis so far is based on the BCJR algorithm that supposes soft information exchange, it was shown in [8] that a hard-input hard-output (HIHO) decoding algorithm (namely the Viterbi algorithm [27]) for convolutional codes is optimal in terms of bit error probability over the BEC. For this reason, we will use a HIHO decoding algorithm for irregular turbo codes inspired by the algorithm in [28] for LDPC codes, in that it propagates in the trellis of the turbo code by removing transitions in the same way edges are removed in a bipartite graph under message-passing decoding [29]. This decoding scheme ensures a decoding complexity linear in the interleaver size. Codes from Table I, although having very high thresholds, suffer from high error floors ( $> 10^{-2}$ ). However, they are well suited for applications for which the quality criterion is the average inefficiency [29]. Fig. 4 shows the performance of irregular turbo codes having a good threshold-error floor tradeoff. In order to avoid high error floors, the 8-state RSC  $(1, 15/13)_8$  code was used as the constituent code of the irregular turbo code. At a target frame error rate of about  $10^{-3}$ , irregular turbo codes are within  $0.018 \leq \Delta_p \leq 0.028$  from capacity for various coding rates.

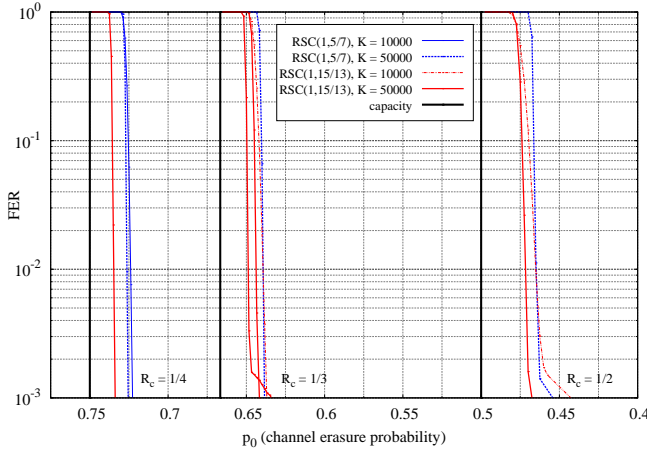


Fig. 4. Performance of irregular turbo codes over the BEC.

## VII. CONCLUSIONS

We proposed irregular turbo codes that perform close to capacity for the binary erasure channel. The codes operate for various coding rates, and they provide low error floors with a PEG-based interleaver that maximizes the cycles in the graphical representation of the code. Implemented with an “on-the-fly” hard-input hard-output decoding algorithm, we believe that these codes are suited for software implementation in upper-layer forward error correction (UL-FEC) contexts.

## REFERENCES

- [1] C. Di, D. Proietti, E. Telatar, T. Richardson, and R. Urbanke, “Finite-length analysis of low-density parity-check codes on the binary erasure channel,” *IEEE Trans. Inf. Theory*, vol. 48, no. 6, pp. 1570–1579, 2002.
- [2] T. Richardson, A. Shokrollahi, and R. Urbanke, “Design of capacity-approaching irregular low-density parity-check codes,” *IEEE Trans. Inf. Theory*, vol. 47, pp. 619–637, 2001.
- [3] M.G. Luby, M. Mitzenmacher, M.A. Shokrollahi, and D.A. Spielman, “Efficient erasure correcting codes,” *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 569–584, 2001.
- [4] H.D. Pfister, I. Sason, and R. Urbanke, “Capacity-achieving ensembles for the binary erasure channel with bounded complexity,” *IEEE Trans. Inf. Theory*, vol. 51, no. 7, pp. 2352–2379, 2005.
- [5] M. Luby, “LT codes,” *Proc. ACM Symp. Found. Comp. Sci.*, 2002.
- [6] A. Shokrollahi, “Raptor codes,” *IEEE/ACM Trans. Networking (TON)*, vol. 14, pp. 2551–2567, 2006.
- [7] B.M. Kurkoski, P.H. Siegel, and J.K. Wolf, “Exact probability of erasure and a decoding algorithm for convolutional codes on the binary erasure channel,” *IEEE GLOBECOM*, 2003.
- [8] B.M. Kurkoski, P.H. Siegel, and J.K. Wolf, “Analysis of convolutional codes on the erasure channel,” *IEEE Int. Symp. Inf. Theory*, 2004.
- [9] E. Rosnes and O. Ytrehus, “Turbo decoding on the binary erasure channel: Finite-length analysis and turbo stopping sets,” *IEEE Trans. Inf. Theory*, vol. 53, pp. 4059–4075, 2007.
- [10] Jeong W. Lee, R. Urbanke, and R.E. Blahut, “On the performance of turbo codes over the binary erasure channel,” *IEEE Comm. Lett.*, vol. 11, pp. 67–69, 2007.
- [11] C. Berrou and A. Glavieux, “Near optimum error correcting coding and decoding: turbo-codes,” *IEEE Trans. Comm.*, vol. 44, pp. 1261–1271, 1996.
- [12] J.J. Boutros, “Asymptotic behavior study of irregular turbo codes,” *7th Int. Work. on DSP Tech. for Space Comm.*, 2001.
- [13] J. Boutros, G. Caire, E. Viterbo, H. Sawaya, and S. Vialle, “Turbo code at 0.03 dB from capacity limit,” *IEEE Int. Symp. Inf. Theory*, 2002.
- [14] M.G. Luby, M. Mitzenmacher, M.A. Shokrollahi, and D.A. Spielman, “Improved low-density parity-check codes using irregular graphs,” *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 585–598, 2001.
- [15] B.J. Frey and D. MacKay, “Irregular turbo codes,” *IEEE Int. Symp. Inf. Theory*, 2000.
- [16] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, “Optimal decoding of linear codes for minimizing symbol error rate,” *IEEE Trans. Inf. Theory*, vol. 20, no. 2, pp. 284–287, 1974.
- [17] S. ten Brink, “Convergence behavior of iteratively decoded parallel concatenated codes,” *IEEE Trans. Comm.*, vol. 49, no. 10, pp. 1727–1737, 2001.
- [18] C. Measson and R. Urbanke, “Further analytic properties of exit-like curves and applications,” *IEEE Int. Symp. Inf. Theory*, 2003.
- [19] R. Storm and K. Price, “Differential evolution: a simple and efficient adaptive scheme for global optimization over continuous spaces,” *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.
- [20] X.-Y. Hu, E. Eleftheriou, and D.M. Arnold, “Regular and irregular progressive edge-growth tanner graphs,” *IEEE Trans. Inf. Theory*, vol. 51-1, pp. 386–398, 2005.
- [21] N. Wiberg, *Codes and Decoding on general graphs*, Ph.D. thesis, Linköping University, Sweden, 1996.
- [22] P.O. Vontobel, “On the construction of turbo code interleavers based on graphs with large girth,” *IEEE ICC*, 2002.
- [23] J. Boutros and O. Pothier, “Convergence analysis of turbo decoding,” *Canadian Work. Inf. Theory*, 1997.
- [24] M. Breiling, “A logarithmic upper bound on the minimum distance of turbo codes,” *IEEE Trans. Inf. Theory*, vol. 50, pp. 1692–1710, 2004.
- [25] R.G. Gallager, *Low density parity check codes*, MIT Press, 1963.
- [26] A. Orlitsky, R. Urbanke, K. Viswanathan, and J. Zhang, “Stopping sets and the girth of Tanner graphs,” *IEEE Int. Symp. Inf. Theory*, 2002.
- [27] A. Viterbi, “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm,” *IEEE Trans. Inf. Theory*, vol. 13, pp. 260–269, 1967.
- [28] M.G. Luby, M. Mitzenmacher, M.A. Shokrollahi, D.A. Spielman, and V. Stemann, “Practical loss-resilient codes,” *Proc. 29th Annual ACM Symp. Theory Comp.*, 1997.
- [29] G. M. Kraidy and V. Savin, “Minimum-delay decoding of turbo-codes for upper-layer fec,” submitted to SPAWC 2008, available on arXiv.org.